

Ham-Sandwich Cuts for Abstract Order Types*

Stefan Felsner[†]

Alexander Pilz[‡]

March 11, 2015

Abstract

The linear-time ham-sandwich cut algorithm of Lo, Matoušek, and Steiger for bi-chromatic finite point sets in the plane works by appropriately selecting crossings of the lines in the dual line arrangement with a set of well-chosen vertical lines. We consider the setting where we are not given the coordinates of the point set, but only the orientation of each point triple (the order type) and give a deterministic linear-time algorithm for the mentioned sub-algorithm. This yields a linear-time ham-sandwich cut algorithm even in our restricted setting. We also show that our methods are applicable to abstract order types.

1 Introduction

Goodman and Pollack investigated ways of partitioning the infinite number of point sets in the plane into a finite number of equivalence classes. To this end they introduced *circular sequences* [17] and *order types* [18]. Two point sets S_1 and S_2 have the same *order type* iff there exists a bijection between the sets s.t. a triple in S_1 has the same orientation (clockwise or counterclockwise) as the corresponding triple in S_2 (in this paper we only consider point sets in general position, i.e., no three points are collinear).

The order type determines many important properties of a point set, e.g., its convex hull and which spanned segments cross. Determining the orientation of a point triple (called a *sidedness query*) can be done in a computationally robust way [4]. Therefore, algorithms that base their decisions solely on sidedness queries allow robust implementations [6]. Furthermore, this restriction is helpful for mechanically proving correctness of algorithms [31, 32]. Another possible advantage of restricting algorithms to sidedness queries allows, for some problems, to cope with degeneracies in point sets. Edelsbrunner and Mücke [11] describe a general framework for the so-called “simulation of simplicity”; in particular, they provide an efficient way to conceptually replace a point set with collinear point triples by one in general position s.t. all other orientations of point triples are preserved.

The duality between point sets and their dual line arrangements is a well-established tool in discrete and computational geometry. Line arrangements can be generalized to pseudo-line arrangements, and many combinatorial and algorithmic questions that can be asked for line arrangements are also interesting for pseudo-line arrangements. The order type of a point set is encoded in the structure of the dual line arrangement of a point set, in particular by the lines (and even by the number of lines [18]) above and below a crossing in the dual arrangement. See [20] for an in-depth treatment of that topic. The orientation of a triple of pseudo-lines can be obtained from the ordering of crossings just as for lines. The triple-orientations fulfill certain axioms, and concepts like the convex hull can be defined for sets with appropriate triple-orientations [22] even though they may not be realizable by a point set. Such a generalization of order types is known as *abstract order type*. Besides their combinatorial properties, algorithmic aspects of abstract order types have been studied. Knuth devotes a monograph [22] to this generalization and its variants, in particular w.r.t. convex hulls. Motivated by Knuth’s open problems, Aichholzer, Miltzow, and Pilz [3] show how to obtain, for a given pair (a, b) of an abstract order type,

*This work has been supported by the ESF EUROCORES programme EuroGIGA - ComPoSe. A.P. is supported by Austrian Science Fund (FWF): I 648-N18. Part of this work has been done while he was recipient of a DOC-fellowship of the Austrian Academy of Sciences at the Institute for Software Technology, Graz University of Technology, Austria. A preliminary version of this paper appeared in the proceedings of ISAAC 2014 [15]. Part of this work was presented in the PhD thesis [33] of the second author.

[†]Institut für Mathematik, Technische Universität Berlin, Germany. felsner@math.tu-berlin.de.

[‡]Institute for Software Technology, Graz University of Technology, Austria. apilz@ist.tugraz.at.

the edges of the convex hull that are intersected by the supporting line of ab in linear time, using only sidedness queries. There appears to be no known reasonable algorithmic problem that can be formulated for both order types and abstract order types such that there is an algorithm for order types that is asymptotically faster than any possible algorithm for abstract order types (see also the discussion in [14, p. 29]). In this paper, we show that the ham-sandwich cut is another problem that does not provide such an example. Apart from being of theoretical interest, abstract order types that are not realizable by point sets occur naturally when the point set is surrounded by a simple polygon and point triples are oriented w.r.t. the geodesics between them [2].

A considerable fraction of problems in computational geometry deals with partitioning finite sets of points by hyperplanes while imposing constraints on both the subsets of the partition as well as on the hyperplanes. In the plane, examples of this class of problems are finding, e.g., a ham-sandwich cut of a bi-chromatic point set [24], a four-way partitioning by orthogonal lines, and a six-way partitioning by three concurrent lines [34], as well as finding three concurrent halving lines that pairwise span an angle of 60° [13].

Given a pair (a, b) of points of a bi-chromatic point set S of n points that are either red or blue, the supporting line of a and b is a *ham-sandwich cut* if not more than half of the red and half of the blue points are on either side of ab . This can be verified by using only sidedness queries (implying a brute-force algorithm running in $\Theta(n^3)$ time). Megiddo [30] presented a linear-time algorithm for the case in which the points of one color are separable from the points of the other color by a line. Edelsbrunner and Waupotitsch [12] gave an $O(n \log(\min\{n_r, n_b\}))$ time algorithm for the general case, with n_r red and n_b blue points. Eventually, a linear-time algorithm was provided by Lo, Matoušek, and Steiger [24] for the general setting (abbrev. *LMS algorithm*). Their approach generalizes to arbitrary dimensions. Bose et al. [7] generalize ham-sandwich cuts to points inside a simple polygon, obtaining a randomized $O((n + m) \log r)$ time algorithm, where m is the number of vertices of the polygon, of which r are reflex. Ham-sandwich cuts belong to a class of problems in computational geometry that deal with partitioning finite sets of points by hyperplanes while imposing constraints on both the subsets of the partition as well as on the hyperplanes; see, e.g., [34] for algorithms for related problems.

The LMS algorithm works on the dual line arrangement of the point set and has to solve the following sub-problem.

Problem 1. *Given a line arrangement \mathcal{A} in the plane and two lines p and q of that arrangement, let v be the vertical line passing through the crossing of p and q . For a subset B of the lines in \mathcal{A} and an integer $k \leq |B|$, find a line $m \in B$ such that the y -coordinate of the point $v \cap m$ is of rank k in the sequence of y -coordinates of the finite point set $v \cap \bigcup_{b \in B} b$.*

This problem can be solved in linear time by directly applying the linear-time selection algorithm [5] to the y -coordinates of the intersections of all lines in B with v . Clearly, the order of the intersections of lines with a vertical line at a crossing is not a property of the order type represented by the arrangement (e.g., in Figure 1, one could change the order of the lines 2 and 4 above the crossing between line 5 and line 6). The order type only determines the set of lines above and below a crossing.

A reformulation of Problem 1 for abstract order types faces two problems. First, the vertical direction is not determined by the order type (this is a property of the circular sequence of the point set); even though we can represent the abstract order type by a pseudo-line arrangement in the Euclidean plane (where there is a vertical direction), there is, in general, an exponential number of different ways to draw a pseudo-line arrangement representing the abstract order type (w.r.t. the x -order of crossings), yielding too many different orders of the pseudo-lines along the vertical line through a crossing. Second, even when such a vertical line is given, directly applying the linear-time selection algorithm

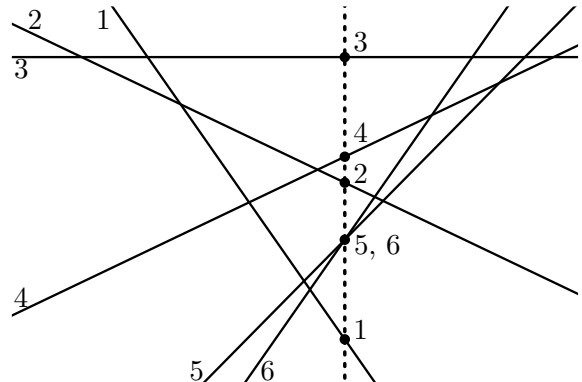


Figure 1: Crossings along a vertical line.

requires that a query comparing the order of two pseudo-lines on the vertical line can be answered in constant time.

In this paper, we show how to overcome these two problems. We define a “vertical” pseudo-line through each crossing in a pseudo-line arrangement and show how to select the pseudo-line of a given rank in the order defined by such a “vertical” pseudo-line. We give the precise definition in Section 2 where we also examine properties of the construction. The result is presented in terms of a (dual) pseudo-line arrangement in the Euclidean plane \mathbb{E}^2 . However, in our model we are not given an explicit representation but are only allowed sidedness queries. In Section 3, we first explain how the queries about a pseudo-line arrangement can be mapped to sidedness queries, and then give a linear-time algorithm for selecting a pseudo-line with a given rank. Our result allows for replacing vertical lines in the LMS algorithm, showing that it also works for abstract order types. An analysis of the LMS algorithm under this aspect is given in Section 4. We obtain

Theorem 1. *A ham-sandwich cut of an abstract order type can be found in linear time using only sidedness queries.*

The observation that the LMS algorithm in principle also works for pseudo-line arrangements has been used by Bose et al. [7] for their randomized linear-time algorithm for geodesic ham-sandwich cuts. However, their pseudo-lines are given by (weakly) x -monotone polygonal paths with a constant number of edges. Hence, the intersection of such a path with a vertical line can be computed in constant time, like in the straight-line setting. Their randomized algorithm runs in $O((n + m) \log r)$ time, where n is the number of red and blue points, m is the number of vertices of the polygon, of which r are reflex. Geodesic order types are a subset of abstract order types [2]. When applying a result from [1] to get, after $O(m)$ preprocessing time, the orientation of each triple of points in a simple polygon in $O(\log r)$ time in combination with the ham-sandwich cut algorithm for abstract order types, we obtain a deterministic $O(n \log r + m)$ time algorithm for geodesic ham-sandwich cuts “for free”. (Note that this does not contradict optimal worst-case behavior shown by Bose et al. [7] for their algorithm, as their analysis is parameterized by $(n + m)$ and r .) We emphasize that a detailed analysis of their approach may give a more fine-grained runtime analysis, and may allow for directly applying common derandomization techniques. Nevertheless, our technique results in a complete separation of the part that is specific to the geodesic setting, implementing a general subroutine, and the ham-sandwich cut algorithm.

A *pseudo-line* is an x -monotone plane curve in \mathbb{E}^2 . A *pseudo-line arrangement* is the cell complex defined by the dissection of \mathbb{E}^2 by a set of pseudo-lines such that each pair of pseudo-lines intersects in exactly one point, at which they cross. An arrangement is *simple* if no three pseudo-lines intersect in the same point. Throughout this paper, let \mathcal{A} be a simple arrangement of n pseudo-lines. The two vertically unbounded cells in \mathcal{A} are called the *north face* and the *south face*. The k -*level* of \mathcal{A} is the set of all points that lie on a pseudo-line of \mathcal{A} and have exactly $k - 1$ pseudo-lines strictly above them. The level of a crossing pq is denoted by $\text{lv}(pq)$ (i.e., pq is separated from the north face by $\text{lv}(pq) - 1$ pseudo-lines). The *upper envelope* of an arrangement is its 1-level, i.e., the union of the segments of pseudo-lines that are incident to the north face.

2 Pseudo-Verticals

It will be convenient to consider all pseudo-lines being directed towards positive x -direction. Let p and q be two pseudo-lines in \mathcal{A} and let p start above q . We denote the latter by $p \prec q$. Our first aim is to define a pseudo-vertical through a crossing, i.e., an object that can be used like a vertical line through a crossing in our abstract setting.

For a crossing pq with $p \prec q$ let γ_{pq} be a curve described by the following local properties. Initially, γ_{pq} passes through the crossing pq and enters the cell C directly above pq ; see Figure 2 (a). To define γ_{pq} it is convenient to think of it as consisting of two parts. The *northbound ray* is the part starting at pq leading to the north face while the *southbound ray* connects pq to the south face. Starting from pq the northbound ray follows p against its direction moved slightly into the interior of cell C . In general the northbound ray of γ_{pq} will be slightly above some line a_i moving against the direction of a_i . When a_i is crossed by a pseudo-line a_j we have two cases. If a_i is crossed from below, γ_{pq} also crosses a_j , and continues following a_i ; see Figure 2 (b). If a_i is crossed by a_j from above, γ_{pq} leaves a_i and continues following a_j against its direction; see Figure 2 (c). This is continued until γ_{pq} follows some line a_i

and all crossings of \mathcal{A} are to the right of the current position along γ_{pq} . At that point, γ_{pq} continues vertically in positive y -direction to infinity (i.e., it crosses all lines a with $a \prec a_i$ in decreasing \prec -order); see Figure 3 (a). The southbound ray of γ_{pq} is defined in a similar manner. It follows some pseudo-lines in their direction but slightly below. It starts with p and when following a_i it changes to a_j at a crossing only when a_j is crossing from above (see Figure 2 (d) and Figure 2 (e)). The final part may again consist of some crossings with lines a with $a \prec a_i$ in decreasing \prec -order. Figure 4 gives an example of a pseudo-vertical in the arrangement of Figure 1, and Figure 5 shows a pseudo-vertical in a wiring diagram.

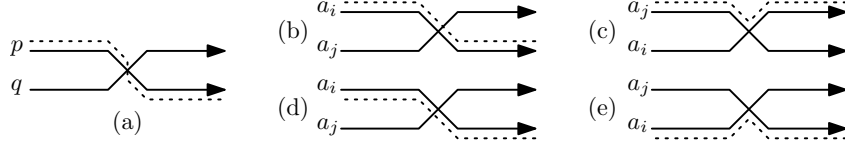


Figure 2: Local definition of a pseudo-vertical γ_{pq} .

We call γ_{pq} a *pseudo-vertical* and, in the following, identify several properties of such a curve. Note that, while we used the (rather informal) notion of “following” a pseudo-line, γ_{pq} is actually defined by the cells it traverses (i.e., two paths in the dual graph of the cell complex starting at the cells above and below pq). As γ_{pq} always follows a pseudo-line of \mathcal{A} or continues in a vertical direction, we note that γ_{pq} is x -monotone.

2.1 Properties of a Pseudo-Vertical

As γ_{pq} always follows a pseudo-line of \mathcal{A} or continues in a vertical direction, we can observe the following.

Observation 1. *For any crossing pq in a pseudo-line arrangement \mathcal{A} , the curve γ_{pq} is x -monotone.*

The following observation can easily be made by visualizing the arrangement as a wiring diagram.

Observation 2. *The number of pseudo-lines above a point moving along γ_{pq} in positive x -direction is a monotone function, it increases at every crossing of γ_{pq} with a pseudo-line of \mathcal{A} .*

Lemma 1. *For any crossing pq in a pseudo-line arrangement \mathcal{A} , the curve γ_{pq} is a pseudo-line such that \mathcal{A} can be extended by γ_{pq} to a new (non-simple) pseudo-line arrangement.*

Proof. Let n be the number of pseudo-lines in \mathcal{A} . Since γ_{pq} continues to vertical infinity in both positive and negative y -direction, it crosses every pseudo-line of \mathcal{A} at least once. From Observation 2, it follows that γ_{pq} crosses at most n pseudo-lines. As γ_{pq} is an x -monotone curve that crosses each pseudo-line of \mathcal{A} exactly once, an extension of \mathcal{A} is again a (non-simple) pseudo-line arrangement. \square

We say that pseudo-line a is above a crossing pq if a is intersected by the northbound ray of γ_{pq} . If a is intersected by the southbound ray of γ_{pq} it is considered to be below pq . (Note that this is equivalent to a separating pq from the north face or the south face, respectively.) Just like a vertical line in a line arrangement, a pseudo-vertical defines a total order on the pseudo-lines of \mathcal{A} by the order it crosses them. We denote the rank of a pseudo-line $m \in \mathcal{A}$ in this order by $\text{rk}_{pq}(m)$. The following lemma shows how we can determine the rank of an element.

Let $L(pq)$ be the set of pseudo-lines in \mathcal{A} such that each $a \in L(pq)$ is below pq and $a \prec p$.

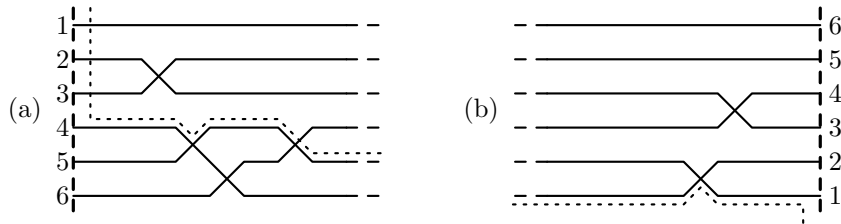


Figure 3: The first (a) and the last (b) pseudo-lines of an arrangement defining γ_{pq} (dotted).

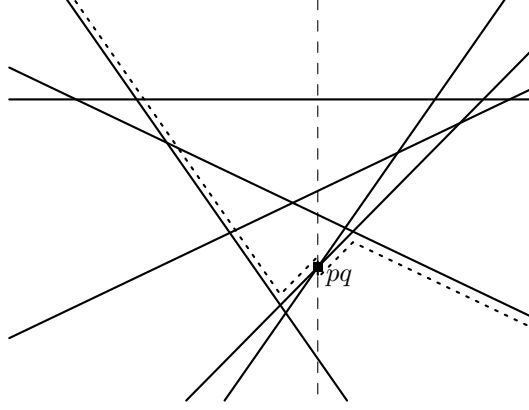


Figure 4: A pseudo-vertical γ_{pq} (dotted) in an arrangement of straight lines.

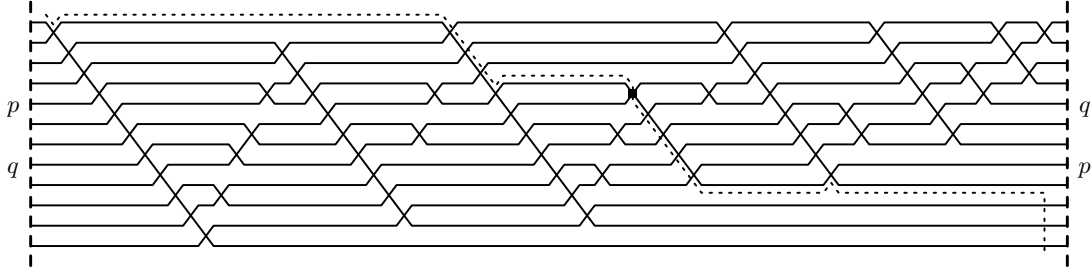


Figure 5: A pseudo-vertical γ_{pq} in a pseudo-line arrangement. The x -order of the crossings represents the order induced by the pseudo-verticals.

Lemma 2. *The northbound ray of γ_{pq} starting from the crossing pq until reaching an unbounded cell for the first time, follows the upper envelope of the sub-arrangement defined by $L(pq) \cup \{p\}$.*

Proof. The proof is by induction on the sequence $\langle p = a_1, a_2, \dots \rangle$ of pseudo-lines that we follow. Clearly, the point pq is on the upper envelope of $L(pq) \cup \{p\}$. Suppose we traverse γ_{pq} in negative x -direction, following a pseudo-line $a_i \in L(pq) \cup \{p\}$. If γ_{pq} crosses a pseudo-line r (i.e., r crosses a_i from below), then r cannot be below pq as it would have to cross γ_{pq} again. If a pseudo-line a_{i+1} crosses a_i from above, then a_{i+1} cannot be above pq as it would have to cross a_i again. Further, γ_{pq} continues on a_{i+1} , keeping the invariant that no element of $L(pq)$ is above the point traversing γ_{pq} . \square

Note that every pseudo-line that passes through the upper envelope (from below) will cross γ_{pq} immediately after that crossing.

Corollary 1. *Let m be a pseudo-line in \mathcal{A} that is above pq and for which there exists a pseudo-line $a \in L(pq) \cup \{p\}$ such that $a \prec m$, i.e., m crosses the upper envelope of $L(pq) \cup \{p\}$ by crossing some $e \in L(pq) \cup \{p\}$ with $e \prec m$. Then the rank $\text{rk}_{pq}(m)$ equals the number of pseudo-lines above the crossing of e and m .*

If m does not intersect the upper envelope of $L(pq) \cup \{p\}$ at some point in negative x -direction of pq , it crosses q before crossing any of the pseudo-lines of $L(pq)$. Therefore, we observe:

Observation 3. *If a pseudo-line m starts above every pseudo-line in $L(pq)$, then the rank of m along γ_{pq} is given by the number of pseudo-lines starting above m increased by 1, i.e., $|\{a \in \mathcal{A} : a \prec m\}| + 1$.*

2.2 Ordering Pseudo-Verticals

Given two different crossings pq and rs in \mathcal{A} , it is easy to see that γ_{pq} and γ_{rs} may follow the same part of a pseudo-line. Nevertheless, one can show that γ_{pq} and γ_{rs} will never intersect when drawn appropriately. See Figure 6 for an illustration accompanying the proof of the following lemma.

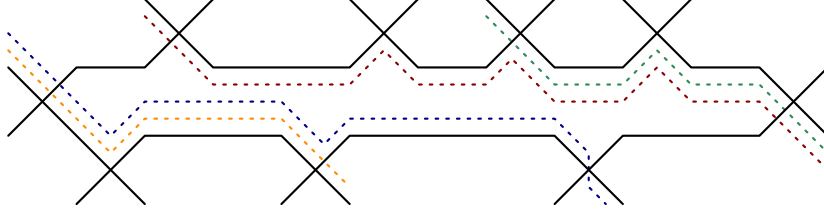


Figure 6: Four pseudo-verticals (two northbound and two southbound parts) meeting in a common cell.

Lemma 3. *The set of pseudo-verticals for all crossings of a pseudo-line arrangement \mathcal{A} can be drawn such that no two pseudo-verticals intersect.*

Proof. Recall that the pseudo-verticals are fully defined by the sequence of cells they traverse. Further, recall that each bounded cell has a unique leftmost and rightmost crossing. For two pseudo-verticals to intersect, they have to enter a common cell C .

Suppose first that C is bounded. Observe that, when traversing, say, γ_{pq} in positive x -direction, then γ_{pq} enters C from above. Let C be between the levels k and $(k+1)$. If the current part of γ_{pq} is northbound, it enters C at the k -level through the pseudo-line defining the leftmost crossing of C (in \mathcal{A}). If it is southbound, it leaves C at the $(k+1)$ -level through the pseudo-line defining the rightmost crossing of C . For two pseudo-verticals to cross inside C , they would have to enter and leave C through four different pseudo-lines (otherwise, we could draw them without crossing in C , probably changing their relative order in the next cell). But this can only happen when one pseudo-vertical is northbound and the other is southbound in that cell (as otherwise they would either enter or leave C through the same pseudo-line), and in that case, there cannot be a crossing inside C , as the pseudo-lines follow the different levels. Once two, say, southbound rays meet in a cell (i.e., they leave a cell through the same pseudo-line of \mathcal{A}), they follow the same pseudo-lines until reaching the south face (i.e., they pass through the same sequence of cells), and hence they can be drawn without intersecting each other.

For unbounded cells, the same argument works, with the exception that along the northbound part a pseudo-vertical enters the cell through the leftmost upper pseudo-line (i.e., at level k), and the southbound part leaves the cell through the rightmost pseudo-line at level $(k+1)$. \square

An augmentation of \mathcal{A} with a complete collection of non-intersecting pseudo-verticals defines a total order on the vertices in the arrangement (cf. the notion of “ P -augmentation” in [19]). Given an arrangement of lines, Edelsbrunner and Guibas [9, 10] define a *topological sweep* as a sweep of an arrangement of lines with a moving curve that intersects each line exactly once. The topological sweep has been generalized to pseudo-line arrangements by Snoeyink and Hershberger [35]. At any point in time during the sweep, the sweeping curve may pass over at least one crossing of the arrangement, maintaining the property that it intersects each line exactly once. However, in contrast to a straight vertical line, there can be several crossings that may be passed next by the sweep curve. It can be observed that we obtain the order of crossings determined by the pseudo-verticals by always sweeping over the lowest-possible crossing in a topological sweep. In the wiring diagram shown in Figure 5, the x -order of the crossings represents this order.

Lemma 3 shows that we can add pseudo-verticals to an arrangement such that they only intersect at vertical infinity. Hence, pseudo-verticals can be considered as a P -augmentation (cf. [19]) of the initial arrangement \mathcal{A} , and we can actually draw a wiring diagram where all the pseudo-verticals can also be represented by vertical lines.

Lemma 4. *The relative order of two pseudo-verticals can be obtained by a linear number of sidedness queries and queries of the form $a \prec b$.*

Proof. For two pseudo-verticals γ_{pq} and γ_{rs} (we have $p \prec q$ and $r \prec s$), determining the relative order means we have to find out whether r crosses s before or after crossing γ_{pq} , i.e., whether rs is to the left or to the right of γ_{pq} . The two pseudo-lines defining a crossing naturally partition the plane into four regions, which we call the upper, lower, left, and right *quadrant* of the crossing. If rs is in the left quadrant of pq (i.e., below p and above q), then rs is definitely to the left of γ_{pq} . Similarly, if rs is in the right quadrant of pq then rs is to the right of γ_{pq} . The analogous holds when exchanging the roles

of pq and rs . Therefore, we can assume without loss of generality that rs is in the upper quadrant of pq (as the other case is symmetric), and that pq is either in the upper or lower quadrant of rs . Consider first the case where pq is in the upper quadrant of rs . If $r \prec p$, then r is part of $L(pq)$, and rs is, by Lemma 2, to the left of γ_{pq} . Analogously, if $p \prec r$, then p is part of $L(rs)$, and therefore rs is to the right of γ_{pq} . We are therefore left with the case where pq is in the lower quadrant of rs . If there exists a pseudo-line $a \in L(pq)$ that is above rs , we again know by Lemma 2 that rs is to the left of γ_{pq} . If no such pseudo-line exists, then $\text{rk}_{pq}(r) < \text{rk}_{pq}(s)$, and therefore, the crossing rs is to the right of γ_{pq} . \square

3 Linear-Time Pseudo-Line Selection

We now discuss algorithmic properties of pseudo-verticals. For the definition of pseudo-verticals and rank we assumed full knowledge about \mathcal{A} . The next task will be to make the notions accessible in the setting where we can only query the abstract order type through an oracle. At the end we aim at using the oracle to select a pseudo-line of given rank w.r.t. a pseudo-vertical in $O(n)$ time.

3.1 An Oracle for an Arrangement

Before answering queries on a pseudo-line arrangement, we need to have an internal representation of the arrangement (without explicitly building it). Let P be a predicate representing an abstract order type on a set S as a counterclockwise oracle, i.e., if there is a primal point set for S then $P(x, y, z)$ tells us whether the three points form a counterclockwise oriented triangle or not, in the general setting P represents a chirotope. From [3] we borrow a linear-time procedure to determine an extreme point x of S using only queries to P . We then use the following internal representation: For all $a \in S \setminus \{x\}$, we define that $x \prec a$. For two points $a, a' \in S \setminus \{x\}$, we define that $a \prec a'$ if and only if $P(x, a, a')$, i.e., in the arrangement the crossing ax precedes $a'x$ on x . For two points $p, q \in S$ with $p \prec q$, the dual pseudo-line r is below the crossing pq if and only if $P(p, q, r)$. Hence, for three points $u, v, w \in S \setminus \{x\}$, the dual line r is below the crossing defined by the (unordered) pair (u, v) if and only if $P(u, v, w) = P(u, v, x)$, i.e., above/below queries for the arrangement of pseudo-lines corresponding to P can be answered in constant time. Note that a constant number of these queries also specify whether the crossing ap precedes the crossing bp on p .

Observe that, with this representation, the first unbounded cell we meet when traversing γ_{pq} against its direction is the north face, because every pseudo-line is crossed by the pseudo-line x from above (see again Figure 5). However, we will not make use of this fact in the remainder of this paper, in particular since we use the fact that the problem is symmetric when exchanging the role of the north face and the south face (which corresponds to rotating the arrangement by 180°).

Our linear-time rank selection algorithm will depend on removing a linear fraction of the pseudo-lines in each iteration. However, the procedure must not remove the extreme point x , to keep the sub-arrangements consistent with the full arrangement.

3.2 Selecting a Pseudo-Line

For a given k , we want to select the pseudo-line m of rank k along γ_{pq} . For a subset B of pseudo-lines and $m \in B$, we denote with $\text{rk}_{pq}(m, B)$ the rank of m within B on γ_{pq} .

In the straight-line version, a linear-time selection algorithm can be used to find an element of rank k in $O(n)$ time. This relies on the fact that the relative position of two lines can be computed in constant time. Comparing $\text{rk}_{pq}(s)$ and $\text{rk}_{pq}(r)$ in the abstract setting can be reduced to deciding whether the crossing rs is below some pseudo-line $a \in L(pq) \cup \{p\}$. Doing this naively results in a linear number of queries and hence we get a selection algorithm with $\Omega(n^2)$ worst-case behavior. We therefore need a more sophisticated method.

When discussing the relative position of two pseudo-verticals, we have seen that checking whether a crossing rs , $r \prec s$, is below γ_{pq} (in which case we have $\text{rk}_{pq}(s) < \text{rk}_{pq}(r)$), may require to determine whether rs is below a pseudo-line $a \in L(pq)$, which, in the worst case, results in a linear number of comparisons.

Let m be the (unknown) pseudo-line of rank k within B . We use a prune-and-search approach to identify m . By counting the elements of B above pq , we determine whether m is above or below pq

(using $O(n)$ queries). Without loss of generality, assume m is above pq (the other case is symmetric) and let U be the set of pseudo-lines above pq . Since removing pseudo-lines from U does not change the structure of the northbound part of γ_{pq} , we can restrict attention to $U_B = U \cap B$. We can also ignore (remove) pseudo-lines below pq that are not in $L(pq)$, i.e., each pseudo-line l below pq such that $p \prec l$.

As a next step, we can, in linear time, verify whether m starts above all pseudo-lines in $L(pq) \cup \{p\}$. If this is the case, the rank of m is determined by the order in which the pseudo-lines start, and we can apply the standard selection algorithm using this order (recall Observation 3).

We are therefore left with the case where m starts below some element $a \in L(pq) \cup \{p\}$. By Corollary 1, we know that we have to find the pseudo-line e where m crosses the upper envelope of $L(pq) \cup \{p\}$ (recall that we have $e \prec m$).

Basically, the algorithm continues as follows. We alternately remove elements in U_B and $L(pq)$ such that the pseudo-lines e and m remain in the respective set until we are left with only a constant number of pseudo-lines in the arrangement. We describe these two pruning steps in two versions, first in a randomized version and after that in a deterministic version. In particular the pruning of U_B turns out to be much simpler in the randomized version.

3.2.1 Randomized Pruning

We first show how to remove pseudo-lines from U_B : Pick uniformly at random a pseudo-line $u \in U_B$. In time proportional to the size of $L(pq)$ we find the last $a \in L(pq) \cup \{p\}$ crossed by u . Since the crossing of u with γ_{pq} is immediately after the crossing with a we can use the crossing ua to split U_B into elements of rank less than $\text{rk}_{pq}(u, U_B)$ and elements of larger rank. One of the two sets can be pruned.

Now we turn to removing pseudo-lines from $L(pq)$: One approach is to consider the k -level σ in the sub-arrangement induced by U_B . We observe that no element of $L(pq)$ can cross σ from below before σ crosses the upper envelope of $L(pq) \cup \{p\}$, as such a pseudo-line of $L(pq)$ would have to cross that element of U_B again before pq . All pseudo-lines in $L(pq)$ that start below σ can therefore be pruned. Among the remaining elements of $L(pq)$, the crossings with σ define a total order. From the remaining elements, pick, uniformly at random, a pseudo-line $b \in L(pq)$ and select (in $O(n)$ time) the pseudo-line $m' \in U_B$ where b crosses σ . We know that m' is unique for the choice of b . We may prune all elements $b' \in L(pq)$ that are below bm' , as no element of $L(pq)$ can cross σ more than once, and hence, no such b' can be e (the pseudo-line where m leaves the upper envelope of $L(pq) \cup \{p\}$). The total order on the remaining elements in $L(pq)$ implies that we can expect half of the elements to be pruned.

With the target of obtaining a deterministic version of our algorithm in mind, we describe the following alternative variant for pruning $L(pq)$. Suppose we are given any crossing vw , with $v, w \in L(pq)$ and $v \prec w$, on the upper envelope of $L(pq) \cup \{p\}$; see Figure 7. Let $\text{lv}(vw)$ be the number of pseudo-lines of U_B above the crossing vw . Depending on the value of $\text{lv}(vw)$, we remove the pseudo-lines of $L(pq)$ that cannot be on the part of the upper envelope that contains the crossing with m : On p consider the crossings vp and wp . Elements of $L(pq)$ that contribute to the upper envelope between vw and pq cross p after wp . Similarly, elements of $L(pq)$ that contribute to the upper envelope between the north face and vw cross p before vp . Hence, depending on $\text{lv}(vw)$, we can remove either the pseudo-lines in $L(pq)$ that cross p before wp or after vp . It remains to choose vw to prune enough points. The median t of the intersections of pseudo-lines from $L(pq)$ with p can be found with a linear number of queries (even deterministically). Based on t , we partition $L(pq)$ into the left part L and the right part R . Find the \prec -minimal element r^* of R and remove all elements $l \in L$ with $r^* \prec l$. The removed elements do not contribute to the upper envelope of $L(pq) \cup \{p\}$. If all elements of L are removed by that, we are done. Otherwise, we want to find the unique crossing vw of a pseudo-line $v \in L$ and a pseudo-line $w \in R$ on the upper envelope of $L(pq)$. Observe that, in the primal, vw corresponds to an edge of the convex hull of $L(pq) \cup \{p\}$ that is stabbed by the supporting line of pt . Finding vw in linear time is described in [3]. For the sake of self-containment, we give a description of the randomized variant of that algorithm in terms of calls to our oracle. We start by picking, uniformly at random, a pseudo-line $r \in R$ and then determine the last crossing rl with a pseudo-line from L on r . It can be argued (cf. Lemma 5) that every pseudo-line $r' \in R$ whose crossing with l is behind the crossing rl fails to be a candidate for w and can hence be removed. We expect to remove half of the pseudo-lines from R through this. A symmetric step can be used to reduce the size of L . By always applying the reduction to the larger of the two we obtain a procedure that outputs the pair vw with expected $O(|L(pq)|)$ queries. Next we determine which elements of U_B are above and which below vw . From this we deduce whether the element $m \in U_B$ with

$\text{rk}_{pq}(m, U_B) = k$ intersects γ_{pq} in the part where it follows the envelope of L or where it follows the envelope of R . Depending on this we can either prune L or R from $L(pq)$. It remains to show that the expected number of queries is in $O(n)$.

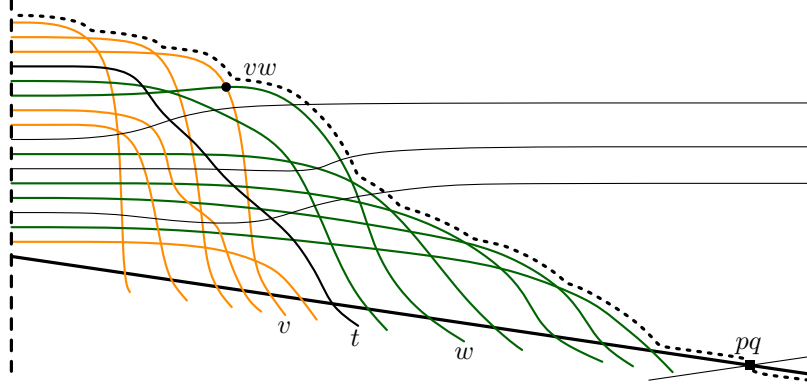


Figure 7: Partitioning the pseudo-lines in $L(pq)$ along p by a pseudo-line t .

We begin with the analysis of the expected number of oracle calls. The justification of some of the claims that are made is in the lemmas below. (They are closely related to arguments used in [3] in the primal, and are stated here in terms of queries to our oracle for the sake of self-containment.)

Theorem 2. *Given an arrangement \mathcal{A} of pseudo-lines, a subset B of its pseudo-lines, a crossing pq , and a natural number $k \leq |B|$, the pseudo-line $m \in B$ with $\text{rk}_{pq}(m, B) = k$, can be found with a randomized algorithm that uses an expected linear number of calls to the oracle representing \mathcal{A} .*

Proof. Deciding whether m is above or below pq and computing the initial sets U_B and $L(pq)$ can be done deterministically with a linear number of queries.

A pruning step for U_B requires $O(|U_B| + |L(pq)|)$ queries. We prune U_B only if $|L(pq)| \leq |U_B|$ and in expectation we prune at least one quarter of the elements of U_B . Therefore, the expected number of queries to select m from B is linear in $|B|$.

A pruning step for $L(pq)$ starts with the median computation on p . This requires $O(|L(pq)|)$ queries in expectation. After that we have the sets L and R and again with a linear number of queries we make sure that $l \prec r$ for all $l \in L$ and $r \in R$. Iteratively prune the larger of L and R with $O(|L| + |R|)$ queries. Upon pruning the expected size of the set halves (Lemma 8). Hence, for pruning steps on L we expect to use $O(|L|)$ queries and symmetrically for R . Finally, we need $O(|U_B|)$ to decide which of L and R can be discarded. Since $|U_B| \leq |L(pq)|$ we conclude that halving the size of $L(pq)$ can be done with $O(|L(pq)|)$ queries. Thence the total number of queries used in the pruning of $L(pq)$ is expected to be in $O(|L(pq)|)$. \square

Lemma 5. *Let l be the last pseudo-line crossing r and let r' be a pseudo-line crossing l after the crossing rl , then $r' \neq w$.*

Proof. Suppose there is a pseudo-line $a \in L$ such that ar' is the crossing on the upper envelope. Then it follows that the crossing lr is above r' and because $a \prec r$ the crossing ar is below r' . This, however implies that on r the crossing with l precedes the crossing with a . This is a contradiction to the choice of l . \square

The dual pruning of L is as follows: Pick uniformly at random a pseudo-line $l \in L$ and determine the first crossing rl with a pseudo-line from R on l . From Lemma 6 we obtain that every pseudo-line $l' \in L$ whose crossing with r precedes the crossing rl fails to be a candidate for v and can hence be removed.

Lemma 6. *Let r be the first pseudo-line crossing l and let l' be a pseudo-line crossing r before the crossing lr , then $l' \neq v$.*

Proof. Suppose there is a pseudo-line $b \in R$ such that $l'b$ is the crossing on the upper envelope. Then it follows that the crossing $l'b$ is above l and because $l \prec r$ the crossing $l'r$ is below l . This, however

implies that on l the crossing with b precedes the crossing with r . This is a contradiction to the choice of r . \square

Lemma 7. *For every pair (l, l') of pseudo-lines in L we have: if l' is not removed when choosing l as the (random) pseudo-line for pruning, then l is removed when l' is chosen as the (random) pseudo-line for pruning.*

Proof. If the same pseudo-line $r \in R$ is the first to cross l and l' , then the statement is obvious. If r is the first to cross l and r' is the first on l' , then there must be a crossing of rr' between lr and lr' on r . Hence on r' the crossing with l precedes the crossing with l' . \square

Lemma 8. *The expected size of the set obtained with a pruning step from L is at most $|L|/2$.*

Proof. On the set L we define a directed graph G_L with edges $l \rightarrow l'$ if l' is removed when choosing l as the (random) pseudo-line for pruning. Lemma 7 implies that G_L is a tournament. The expected number of pseudo-lines removed in the pruning equals the expected out-degree plus one. The precise value is $(|L| + 1)/2$ so in expectation less than $|L|/2$ pseudo-lines remain. \square

The analogous statements of Lemma 7 and Lemma 8 for the set R are true as well.

Remark. Note that, by removing pseudo-lines from $L(pq)$, we obtain a new arrangement \mathcal{A}' , in which the pseudo-vertical γ_{pq} will, in general, follow different pseudo-lines from $L(pq)$ along its northbound ray. Still, the number of pseudo-lines above the crossing em that we look for remains the same, and m will have the same rank with respect to the new pseudo-vertical γ'_{pq} in \mathcal{A}' .

3.2.2 Deterministic Pruning

To remove pseudo-lines from $L(pq)$ deterministically, we can directly apply the deterministic algorithm given in [3] to find vw after $O(n)$ queries.

Recall that to remove elements of U_B , we pick a pseudo-line $u \in U_B$. We compute the rank $\text{rk}_{pq}(u, B)$ by finding the corresponding pseudo-line $b \in L(pq) \cup \{p\}$ at which u passes through the upper envelope of $L(pq) \cup \{p\}$. Clearly, this can be done in linear time using our basic operations. If $u = m$, we are done. If $\text{rk}_{pq}(u, B) < k$, then all pseudo-lines in U_B below bu can be removed (we will see how to choose u to remove a constant fraction of the pseudo-lines). Otherwise, we remove all pseudo-lines in U_B above bu and update k accordingly. While by this operation we obtain a new arrangement \mathcal{A}' , the northbound ray of γ_{pq} in \mathcal{A}' is defined by the same pseudo-lines as in \mathcal{A} , and we can therefore safely continue with the next iteration. It remains to show how to pick u in a deterministic way such that at least a constant fraction of U_B can be removed. To this end, we use the concept of ε -approximation of range spaces.

Our definitions follow [28]. A *range space* is a pair $\Sigma = (X, \mathcal{R})$ where X is a set and \mathcal{R} is a set of subsets of X . The elements of \mathcal{R} are called *ranges*. For X being finite, a subset $A \subseteq X$ is an ε -approximation for Σ if, for every range $R \in \mathcal{R}$, we have

$$\left| \frac{|A \cap R|}{|A|} - \frac{|X \cap R|}{|X|} \right| \leq \varepsilon .$$

A subset Y of X is *shattered* by \mathcal{R} if every possible subset of Y is a range of Y . The *Vapnik-Chervonenkis dimension* (VC-dimension) of Σ is the maximum size of a shattered subset of X . For sets with finite VC-dimension, Vapnik and Chervonenkis [36] give the following seminal result.

Theorem 3 (Vapnik, Chervonenkis [36]). *Any range space of VC-dimension d admits an ε -approximation of size $O(d/\varepsilon^2 \log(d/\varepsilon))$.*

For $|X| = n$, the *shatter function* $\pi_{\mathcal{R}}(n)$ of a range space (X, \mathcal{R}) is defined by

$$\pi_{\mathcal{R}}(n) = \max\{|\{Y \cap R : R \in \mathcal{R}\}| : Y \subseteq X\} .$$

Vapnik and Chervonenkis [36] show that, for a range space (X, \mathcal{R}) of VC-dimension d , $\pi_{\mathcal{R}}(n) \in O(n^d)$ holds. Matoušek [27, 29] gives, for a constant ε , a linear-time algorithm for computing a constant-size ε -approximation for range spaces of finite VC-dimension d (simplified by Chazelle and Matoušek [8]), provided there exists a subspace oracle.

Definition 1. A subspace oracle for a range space (X, \mathcal{R}) is an algorithm that returns, for a given subset $Y \subseteq X$, the set of all distinct intersections of Y with the ranges in \mathcal{R} , i.e., the set $\{Y \cap R : R \in \mathcal{R}\}$ and runs in time $O(|Y| \cdot h)$, where h is the number of sets returned.

Theorem 4 (Matoušek [27, Theorem 4.1]). Let $\Sigma = (X, \mathcal{R})$ be a range space with the shatter function $\pi_{\mathcal{R}}(n) \in O(n^d)$, for a constant $d \geq 1$. Given a subspace oracle for Σ and a parameter $r \geq 2$, a $(1/r)$ -approximation for Σ of size $O(r^2 \log r)$ can be computed in time $O(|X|(r^2 \log r)^d)$.

Observe that, for such a range space, the running time of the subspace oracle is bounded by $O(|Y|^{d+1})$, as h is at most $\pi_{\mathcal{R}}(|Y|)$.

Suppose that, e.g., X is a point set in the Euclidean plane and \mathcal{R} consists of all possible subsets of X defined by half-planes, defining a range space $\Sigma = (X, \mathcal{R})$. Hence, \mathcal{R} is the set of *semispaces* defined by the order type of X . The VC-dimension of (X, \mathcal{R}) is known to be 3 [28]. Hence, a constant-size ε -approximation of a point set for \mathcal{R} exists. (As pointed out in [24], this approximation allows constructing an approximate ham-sandwich cut in constant time, such that on every side of the cut there are no more than $1/2 + \varepsilon$ of the points of each class.) The subspace oracle returns, for any subset Y of points, all possible ways a line can separate Y , which can easily be done in time $O(|Y|^3)$.

The VC-dimension of 3 for that range space holds also for abstract order types (see also [16]). A subspace oracle for semispaces of a given set can easily be implemented using the definition of a semispace by allowable sequences [19]; for each pair $f, g \in Y, f \prec g$, in the dual pseudo-line arrangement, we report the pseudo-lines above the crossing fg and, say, f as a semispace, and the pseudo-lines below fg and g as a second semispace.

Consider again the set U_B of pseudo-lines above the crossing pq in \mathcal{A} . Using Theorem 4, we obtain an ε -approximation $A \subset U$ for the range space of semispaces, i.e., the pseudo-lines of U above and below a point in \mathcal{A} . A is of constant size for a fixed ε . For each pseudo-line $o \in A$, we obtain the crossing of o with the pseudo-lines in $L(pq)$ that defines the rank $\text{rk}_{pq}(o)$ in $O(n)$ time. Let $u_A \in A$ have the median rank among the elements of A . Then not less than $1/2 - \varepsilon$ pseudo-lines of U are above and below the crossing bu_A on the upper envelope of $L(pq)$; we may prune a constant fraction of the elements in U_B .

3.2.3 Analysis

In each iteration, our problem consists of the remaining pseudo-lines in U and in $L(pq)$, plus a constant number of additional pseudo-lines (i.e., p, q , and the pseudo-lines needed by the oracle, in our case the pseudo-line x). Let n be the number of these pseudo-lines. In each iteration we prune the larger of U and $L(pq)$. In both cases, we remove at least half of the pseudo-lines on one side of pq , and therefore $n/4 - O(1)$ pseudo-lines in each iteration. Since each iteration takes $O(n)$ time, we have overall a linear-time prune-and-search algorithm.

Theorem 5. Given an arrangement \mathcal{A} of pseudo-lines, a subset B of its pseudo-lines, a crossing pq , and a natural number $k \leq |B|$, the pseudo-line $m \in B$ of rank k in B on the pseudo-vertical through pq , i.e., $\text{rk}_{pq}(m, B)$, can be found in linear time using only sidedness queries on the corresponding abstract order type.

Remark. Several years before the linear-time algorithm for ham-sandwich cuts [24] was developed, Megiddo [30] considered the following restricted version of the ham-sandwich cut problem. Given a set of red and a set of blue points with disjoint convex hulls, find a line that bisects both the red and the blue point set. Actually, the resulting line does not have to be a bisector, but the number of red and blue points on one side of the line can be chosen arbitrarily. In the dual representation, we are given m blue lines with positive slope and n red lines with negative slope, and we want to find the intersection point between a k_1 -level in the blue lines and the k_2 -level in the red lines. If we consider the pseudo-lines in $L(pq) \cup \{p\}$ as red pseudo-lines and the pseudo-lines in U as the blue ones, we are looking for the intersection point between the k -level in U and the 1-level in $L(pq) \cup \{p\}$. However, Megiddo's algorithm also depends on the realization of the line arrangement; the algorithm requires selecting the median of a subset of crossings ordered by their x -coordinate and selecting the intersections of a given rank at vertical lines. These problems also have to be solved when abstracting the general ham-sandwich cut algorithm.

4 Revisiting the Ham-Sandwich Cut Algorithm

In this section, we describe an application of pseudo-verticals for a bisection algorithm, namely the linear-time ham-sandwich cut algorithm by Lo, Matoušek, and Steiger [24] (the LMS algorithm). To this end, we revisit the description given in [24]; we adapt some of the terminology (like replacing “line” with “pseudo-line”) and argue for the correspondence between entities in the original description and their abstract counterpart.

Lo, Matoušek, and Steiger [24] describe two different variants of the algorithm, one for points in the plane and the other one for points in arbitrary dimension (their work is a generalization of a 2-dimensional version by Lo and Steiger presented in [25]). For the 2-dimensional case, a result by Matoušek [26, Theorem 3.2] is used for appropriately selecting a set of vertical lines. In higher dimensions, they use a different approach (given in [27]) based on an ε -approximation with the ranges being defined as sets of hyperplanes that are stabbed by segments (we will give a formal definition later). While the higher-dimensional variant appears to be less instructive, it is easier to apply to our setting. We therefore will use this variant for our 2-dimensional setting; here, we do not give the description for arbitrary dimension, but transcribe it to dimension 2 only. Still, our exposition closely follows [24], while merely pointing out the parts where the applicability to our abstract setting might not be obvious.

Let P be a finite set of n points in the Euclidean plane. A line h *bisects* P if no more than $n/2$ points lie in either of the open half-planes defined by h . We call h a *bisector*. If P is a disjoint union of two point sets P_1, P_2 , a *ham-sandwich cut* is a line that simultaneously bisects both P_1 and P_2 (a *red* and a *blue* set). This definition extends to abstract order types in a natural way. It is well-known that a ham-sandwich cut always exists. Let T be an interval on the x -axis, and let $V(T)$ be the vertical slab between the two vertical lines defining T . The interval has the *odd intersection property* with respect to the levels λ_1 and λ_2 if $|(\lambda_1 \cap \lambda_2) \cap V(T)|$ is odd. If $k = \lfloor (n+1)/2 \rfloor$, the k -level is called *median level*. In our case, each slab is defined by two pseudo-verticals.

The algorithm works in a prune-and-search manner. Let us first consider the setting where we are given an actual set of points in \mathbb{E}^2 . In every iteration, we are given

- an interval T on the x -axis,
- two sets G_1 and G_2 of lines dual to a subset of points in P_1 and P_2 , respectively, with $|G_1| = n_1$ and $|G_2| = n_2$, and
- two integers k_1 and k_2 , with $1 \leq k_1 \leq n_1$ and $1 \leq k_2 \leq n_2$, denoting the k_1 -level λ_1 and the k_2 -level λ_2 , respectively.

Further, we know that T has the odd intersection property for the k_1 -level and the k_2 -level. We denote the arrangements corresponding to G_1 and G_2 with \mathcal{A}_1 and \mathcal{A}_2 , respectively. Initially, λ_1 and λ_2 are the median levels of the two arrangements (for which the odd intersection property holds). Without loss of generality, suppose $n_1 \geq n_2$. The algorithm consists of the following four steps:

1. Divide T into a constant number of subintervals T_1, \dots, T_C , to limit the number of pseudo-lines that are on λ_1 within each subinterval.
2. Find a subinterval T_j with the odd intersection property.
3. Construct a trapezoid $\tau \subset V(T_j)$ such that
 - (a) $\lambda_1 \cap V(T_j) \subset \tau$, and
 - (b) at most half of the lines of \mathcal{A}_1 intersect τ .
4. Discard the lines of \mathcal{A}_1 that do not intersect τ , update k_1 accordingly and continue within the interval T_j .

In our abstract setting, the interval T is given by a pair of pseudo-verticals. Recall that there is a total order on the pseudo-verticals of a pseudo-line arrangement. The trapezoid τ will also be replaced by a corresponding structure that will be described later.

4.1 Obtaining Intervals

Step 1 in the algorithm is the one that is technically most involved. The straight-line version can be solved using the following result by Matoušek.¹

Lemma 9 (Matoušek). *Let H be a collection of n hyperplanes in \mathbb{E}^d and let \mathcal{R} be all subsets of H of the form $\{h \in H : h \cap s \neq \emptyset\}$ where s is a segment in \mathbb{E}^d . An ε -approximation for the range space (H, \mathcal{R}) of size $O(\varepsilon^{-2} \log(1/\varepsilon))$ can be computed in time $O(f(\varepsilon)n)$, where $f(\varepsilon)$ is a factor depending on ε and d only.*

Let us go into the details why this lemma also holds for arrangements of pseudo-lines. To this end, a general result by Haussler and Welzl [21] is used.

Lemma 10 (Haussler, Welzl [21, Lemma 4.5]). *Assume $k \geq 1$ and (X, \mathcal{R}) is a range space of VC-dimension $d \geq 2$. Let \mathcal{R}' be the set of all sets of the form $\bigcup_{i=1}^k R_i - \bigcap_{i=1}^k R_i$, where R_i is a range in \mathcal{R} , $1 \leq i \leq k$. Then (X, \mathcal{R}') has VC-dimension less than $2dk \log(dk)$.*

We already discussed that a range space defined by the semispaces of an abstract order type has VC-dimension 3. We can combine this fact with Lemma 10 in the following way. Consider two semispaces S_1 and S_2 of an abstract order type, defined by the pseudo-lines above two points p_1 and p_2 in the corresponding pseudo-line arrangement \mathcal{A} (for simplicity, suppose that none of p_1 and p_2 lie on a pseudo-line of \mathcal{A}). Let $R = (S_1 \cup S_2) \setminus (S_1 \cap S_2)$. Then R consists exactly of the pseudo-lines that separate p_1 from p_2 . By the Levi Enlargement Lemma (see [23]), we can extend \mathcal{A} by a pseudo-line through any two points, and therefore obtain a pseudo-line χ for \mathcal{A} containing both p_1 and p_2 . Consider the part of χ between p_1 and p_2 . We call such a part a *pseudo-segment*. The pseudo-lines crossed by this pseudo-segment are exactly those in R . Applying Lemma 10, we can therefore obtain a range space that is defined by the pseudo-lines that can be crossed by pseudo-segments from the range space defined by the semispaces; this new range space has again finite VC-dimension. (Note that, while we explained the application of Lemma 10 using points p_1 and p_2 , the argument also holds for pseudo-segments defined by crossings of \mathcal{A} , as the endpoints of the pseudo-segment can be perturbed to be in one of the four cells adjacent to a crossing.) Using Matoušek's linear-time algorithm for obtaining an ε -approximation [27, Theorem 4.1], we can state the following counterpart to Lemma 9 for abstract order types in the plane.

Corollary 2. *Let $\Sigma = (X, \mathcal{R})$ be a range space where X is the set of pseudo-lines in a pseudo-line arrangement \mathcal{A} and \mathcal{R} consists of the sets of pseudo-lines of \mathcal{A} that are crossed by pseudo-segments obtained on \mathcal{A} . Then a $(1/r)$ -approximation of constant size for Σ can be computed in $O(|X|)$ time for a given $r \geq 2$.*

For future reference, let us call this range space the *pseudo-segment range space* of the arrangement.

For the ham-sandwich cut algorithm, we can now proceed in the following way. Using Corollary 2, we obtain an ε -approximation A for the pseudo-segment range space of \mathcal{A}_1 ; we choose $\varepsilon = 1/12$ with foresight. Sort the crossings of A by the order implied by the pseudo-verticals through the crossings on the original arrangement \mathcal{A} . Since A is of constant size, this can be done in $O(|A|)$ time. We use Theorem 5 to determine the k_1 -level of \mathcal{A}_1 at the pseudo-vertical of each crossing in A . Hence, for each pseudo-vertical, we get a crossing in \mathcal{A} that has level k_1 in \mathcal{A}_1 . Counting the elements of \mathcal{A}_2 above each such crossing allows us to find a crossing pq and a crossing $p'q'$ consecutive in A with the odd intersection property. We again use Theorem 5 to select the pseudo-lines that are of rank $k_1 - c\varepsilon n_1$ and $k_1 + c\varepsilon n_1$ in \mathcal{A}_1 at the pseudo-vertical γ_{pq} , and do the same at $\gamma_{p'q'}$ for a constant c (we fix $c = 3/2$ with foresight). Hence, we have six crossings in \mathcal{A} of which we know the level within \mathcal{A}_1 . Let g_l and g_r be the crossings at the k_1 -level along γ_{pq} and $\gamma_{p'q'}$, respectively. We denote the crossings at the $(k_1 - c\varepsilon n_1)$ -level by d_l^- and d_r^- . Their counterparts at the $(k_1 + c\varepsilon n_1)$ -level are denoted by d_l^+ and d_r^+ .

4.2 Properties of a Trapezoid-Like Structure

In the original LMS algorithm [24], the points at the given levels were determined by the intersections of the levels with the vertical lines. These points formed a trapezoid. However, the actual properties

¹Lo, Matoušek, and Steiger [24] refer to [26], where [21, Lemma 4.5] (Lemma 10 herein) is used, and also refer to [27] in this context, where a general algorithm for constructing ε -approximations is given.

used are the ones of the points and not the ones of the trapezoid as a geometric object. In this part, we reproduce the line of arguments used in [24] to show that at least half of the pseudo-lines in \mathcal{A}_1 are either above both d_1^- and d_r^- or below both d_1^+ and d_r^+ , and that these pseudo-lines are not on the k_1 -level between g_l and g_r .

Consider the arrangement \mathcal{A}_1 . We bound the number of pseudo-lines that separate d_1^- from d_r^- , i.e., the pseudo-lines crossing a pseudo-segment between d_1^- and d_r^- . The levels of d_1^- and d_r^- are the same. Therefore, the numbers of pseudo-lines of the approximation A above these two crossings differ by at most $2\varepsilon|A|$. If there would be more than $2\varepsilon|A|$ pseudo-lines of A separating d_1^- from d_r^- , then at least one of these pseudo-lines would have to be above d_1^- and below d_r^- , and another one would have to be below d_1^- and above d_r^- . Hence, the crossing between these two pseudo-lines would have to be in the interval between γ_{pq} and $\gamma_{p'q'}$. But this contradicts the choice of γ_{pq} and $\gamma_{p'q'}$, as there is no pseudo-vertical through a crossing of two pseudo-lines of A between them. Hence, any pseudo-segment between d_1^- and d_r^- crosses at most $2\varepsilon|A|$ of the pseudo-lines in A . By the ε -approximation property, at most $3\varepsilon n_1$ pseudo-lines of \mathcal{A}_1 intersect such a pseudo-segment.

Suppose there is a pseudo-line w of \mathcal{A}_1 that is above both d_1^- and d_r^- , but still w is an element of the k_1 -level between g_l and g_r . The part of the arrangement where this can happen is bounded by γ_{pq} and $\gamma_{p'q'}$. Then also any pseudo-segment s between d_1^- and d_r^- would have to cross the relevant part of the k_1 -level (recall that the pseudo-segment can be considered as a part of a pseudo-line in an extended arrangement). At both d_1^- and d_r^- , the pseudo-segment s is at level $(k_1 - c\varepsilon n_1)$, and therefore has to cross $2(k_1 - (k_1 - c\varepsilon n_1)) = 2c\varepsilon n_1$ pseudo-lines to reach the k_1 -level and then return to level $(k_1 - c\varepsilon n_1)$. By the choice of c , this is exactly $3\varepsilon n_1$, the maximum number of crossings the pseudo-segment s can have. Hence, s cannot go below the k_1 -level and therefore the pseudo-line w cannot intersect the k_1 -level. For our prune-and-search approach, we can therefore remove all pseudo-lines of \mathcal{A}_1 that are above both d_1^- and d_r^- , and, by symmetric arguments, can do the same for the ones below both d_1^+ and d_r^+ .

It remains to count how many pseudo-lines are removed. There are exactly $2c\varepsilon n_1$ pseudo-lines separating d_1^+ from d_1^- , as well as d_r^+ from d_r^- . Further, we argued that there are at most $3\varepsilon n_1$ pseudo-lines between d_1^- and d_r^- , as well as between d_1^+ and d_r^+ . This amounts to $(4c + 6)\varepsilon n_1 = 12\varepsilon n_1$, where each pseudo-line is counted twice. Therefore, we have to keep at most $6\varepsilon n_1 = n_1/2$ pseudo-lines of \mathcal{A}_1 .

4.3 A Note on the Intervals

After having pruned a linear fraction of pseudo-lines, the algorithm performs another iteration within a smaller interval for which the odd-intersection property holds. Note that we need to continue within this interval, as the k_1 -level (for an updated k_1) equals the median-level only within that region. In the geometric variant, the interval was explicitly given by the vertical lines of the current slab. For pseudo-verticals, we have no such fixed position, and actually the pseudo-verticals will, in general, be different when pseudo-lines are removed from the arrangement (even the relative order of two crossings may change). However, we can safely define the interval for the subproblem by the two crossings g_l and g_r , as the odd intersection property can be seen as a property of two points on one of the two levels (only the number of pseudo-lines of \mathcal{A}_2 above each of the two points is relevant here). It is interesting to observe that also γ_{g_l} and γ_{g_r} do not have a different relative position in the new arrangement.

With the proof of Theorem 5 and the observations implying that all the remaining parts of the LMS algorithm actually do not depend on the geometric realization, we obtain Theorem 1 as our main result.

5 Conclusion

In this paper, we defined a possible replacement of a vertical line in line arrangements for arrangements of pseudo-lines and showed that it fulfills important algorithmic properties. In particular, we were able to show how to select the k th pseudo-line crossed by this pseudo-vertical line and the crossing where this pseudo-line (locally) enters the k -level in linear time, using only sidedness queries. As an application, we showed how these pseudo-vertical lines replace vertical lines in the linear-time ham-sandwich cut algorithm by Lo, Matoušek, and Steiger [24].

In essence, the order of the pseudo-verticals through all crossings of a pseudo-line arrangement fix one specific allowable sequence for an abstract order type. Theorem 5 allows us to select certain elements of a

permutation in that allowable sequence in linear time. We have seen that this approach is a generalization of the result presented in [3]. (Note that the oracle also uses the extreme point x , which, in turn, requires applying Theorem 5; any internal representation that can answer queries of the form $a \prec b$ in constant time allows us to find an extreme point in linear time.)

Compared to $L(pq)$, pruning U_B deterministically required the technically involved step of selecting an ε -approximation. Is there a more “light-weight” deterministic way to prune U_B , similar to the one used for $L(pq)$?

References

- [1] O. Aichholzer, T. Hackl, M. Korman, A. Pilz, and B. Vogtenhuber. Geodesic-preserving polygon simplification. In *Proc. 24th International Symposium on Algorithms and Computation (ISAAC 2013)*, volume 8283 of *LNCS*, pages 11–21. Springer, 2013.
- [2] O. Aichholzer, M. Korman, A. Pilz, and B. Vogtenhuber. Geodesic order types. In *Proc. 18th International Computing and Combinatorics Conference (COCOON 2012)*, volume 7434 of *LNCS*. Springer, 2012.
- [3] O. Aichholzer, T. Miltzow, and A. Pilz. Extreme point and halving edge search in abstract order types. *Comput. Geom.*, 46(8):970–978, 2013.
- [4] F. Avnaim, J.-D. Boissonnat, O. Devillers, F. P. Preparata, and M. Yvinec. Evaluating signs of determinants using single-precision arithmetic. *Algorithmica*, 17(2):111–132, 1997.
- [5] M. Blum, R. W. Floyd, V. Pratt, R. L. Rivest, and R. E. Tarjan. Time bounds for selection. *J. Comput. System Sci.*, 7(4):448–461, 1973.
- [6] J.-D. Boissonnat and J. Snoeyink. Efficient algorithms for line and curve segment intersection using restricted predicates. *Comput. Geom.*, 16(1):35–52, 2000.
- [7] P. Bose, E. D. Demaine, F. Hurtado, J. Iacono, S. Langerman, and P. Morin. Geodesic ham-sandwich cuts. In *Proc. 20th Symposium on Computational Geometry (SoCG 2004)*, pages 1–9. ACM, 2004.
- [8] B. Chazelle and J. Matoušek. On linear-time deterministic algorithms for optimization problems in fixed dimension. *J. Algorithms*, 21(3):579–597, 1996.
- [9] H. Edelsbrunner and L. J. Guibas. Topologically sweeping an arrangement. *J. Comput. Syst. Sci.*, 38(1):165–194, 1989.
- [10] H. Edelsbrunner and L. J. Guibas. Corrigendum: Topologically sweeping an arrangement. *J. Comput. Syst. Sci.*, 42(2):249–251, 1991.
- [11] H. Edelsbrunner and E. P. Mücke. Simulation of simplicity: a technique to cope with degenerate cases in geometric algorithms. *ACM Trans. Graph.*, 9(1):66–104, 1990.
- [12] H. Edelsbrunner and R. Waupotitsch. Computing a ham-sandwich cut in two dimensions. *J. Symb. Comput.*, 2(2):171–178, 1986.
- [13] J. Erickson, F. Hurtado, and P. Morin. Centerpoint theorems for wedges. *Discrete Math. Theor. Comput. Sci.*, 11(1):45–54, 2009.
- [14] J. G. Erickson. *Lower Bounds for Fundamental Geometric Problems*. PhD thesis, University of California at Berkeley, 1996.
- [15] S. Felsner and A. Pilz. Ham-Sandwich Cuts for Abstract Order Types. In *Proc. 25th International Symposium on Algorithms and Computation (ISAAC 2014)*, volume 8889 of *LNCS*, pages 726–737. Springer, 2014.
- [16] B. Gärtner and E. Welzl. Vapnik-Chervonenkis dimension and (pseudo-)hyperplane arrangements. *Discrete Comput. Geom.*, 12:399–432, 1994.

- [17] J. E. Goodman and R. Pollack. On the combinatorial classification of nondegenerate configurations in the plane. *J. Comb. Theory, Ser. A*, 29(2):220–235, 1980.
- [18] J. E. Goodman and R. Pollack. Multidimensional sorting. *SIAM J. Comput.*, 12(3):484–507, 1983.
- [19] J. E. Goodman and R. Pollack. Semispaces of configurations, cell complexes of arrangements. *J. Combin. Theory Ser. A*, 37(3):257–293, 1984.
- [20] J. E. Goodman and R. Pollack. Allowable sequences and order types in discrete and computational geometry. In J. Pach, editor, *New Trends in Discrete and Computational Geometry*, pages 103–134. Springer, 1993.
- [21] D. Haussler and E. Welzl. ε -nets and simplex range queries. *Discrete Comput. Geom.*, 2:127–151, 1987.
- [22] D. E. Knuth. *Axioms and Hulls*, volume 606 of *LNCS*. Springer-Verlag, 1992.
- [23] F. Levi. Die Teilung der projektiven Ebene durch Gerade oder Pseudogerade. *Ber. Math.-Phys. Kl. Sächs. Akad. Wiss. Leipzig*, 78:256–267, 1926. In German.
- [24] C.-Y. Lo, J. Matoušek, and W. Steiger. Algorithms for ham-sandwich cuts. *Discrete Comput. Geom.*, 11:433–452, 1994.
- [25] C.-Y. Lo and W. Steiger. An optimal time algorithm for ham-sandwich cuts in the plane. In *Proc. 2nd Canadian Conference on Computational Geometry (CCCG 1990)*, pages 5–9, 1990.
- [26] J. Matoušek. Construction of ε -nets. *Discrete Comput. Geom.*, 5:427–448, 1990.
- [27] J. Matoušek. Approximations and optimal geometric divide-and-conquer. In *Proc. 23rd Symposium on the Theory of Computing (STOC 1991)*, pages 505–511. ACM, 1991.
- [28] J. Matoušek. Epsilon-nets and computational geometry. In J. Pach, editor, *New Trends in Discrete and Computational Geometry*, pages 69–89. Springer, 1993.
- [29] J. Matoušek. Approximations and optimal geometric divide-and-conquer. *J. Comput. System Sci.*, 50(2):203–208, 1995.
- [30] N. Megiddo. Partitioning with two lines in the plane. *J. Algorithms*, 6(3):430–433, 1985.
- [31] L. I. Meikle and J. D. Fleuriot. Mechanical theorem proving in computational geometry. In *Automated Deduction in Geometry*, volume 3763 of *LNCS*, pages 1–18. Springer, 2004.
- [32] D. Pichardie and Y. Bertot. Formalizing convex hull algorithms. In *Proc. 14th International Conference on Theorem Proving in Higher Order Logics (TPHOLs 2001)*, volume 2152 of *LNCS*, pages 346–361. Springer, 2001.
- [33] A. Pilz. *On the Complexity of Problems on Order Types and Geometric Graphs*. PhD thesis, Graz University of Technology, 2014.
- [34] S. Roy and W. Steiger. Some combinatorial and algorithmic applications of the Borsuk–Ulam theorem. *Graphs Combin.*, 23(1):331–341, 2007.
- [35] J. Snoeyink and J. Hershberger. Sweeping arrangements of curves. In *Proc. 5th Symposium on Computational Geometry (SoCG 1989)*, pages 354–363, 1989.
- [36] V. N. Vapnik and A. Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory Probab. Appl.*, 16:264–280, 1971.